IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Application No.: 09/663,563	§	Examiner: Kang, Insun
Filed: September 15, 2000	§	Group/Art Unit: 2193
Inventors:	§	Atty. Dkt. No: 5181-72000
Thomas E. Saulpaugh, et al.	§	
	§	
	§	
Title: Transformation of Objects Between a Computer Programming Language and a Data Representation Language	§	
	§	

PRE-APPEAL BRIEF REQUEST FOR REVIEW

Mail Stop AF
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

Applicants request review of the rejection in the above-identified application. No amendments are being filed with this request. This request is being filed with a notice of appeal. Review is requested for the reasons stated below.

Claims 1-90 are pending in the application. Reconsideration of the present case is earnestly requested in light of the following remarks. Please note that for brevity, only the primary arguments directed to the independent claims are presented, and that additional arguments, e.g., directed to the subject matter of the dependent claims, will be presented if and when the case proceeds to Appeal.

The Examiner rejected claims 1-90 under 35 U.S.C. § 102(e) as being anticipated by Tuatini (U.S. Publication 2002/0035645). Applicants respectfully traverse this rejection for at least the following reasons.

The rejection is improper because Tuatini is not a prior art reference. The Examiner has the burden of proof to produce the factual basis for the rejection. *In re Warner*, 154 USPQ 173, 177 (C.C.P.A. 1967), *cert. denied*, 389 U.S. 1057 (1968). Since the Examiner has not proven that Tuatini qualifies as a prior art reference, the Examiner

has not met this burden of proof and the rejection is improper. More specifically, the Tuatini patent was filed on December 28, 2000, after Applicants' filing date of September 15, 2000. Tuatini does claim the benefit of a provisional application filed December 30, 1999. However, the December 30, 1999 filing date can only be used as Tuatini's prior art date for the subject matter that is **common** to both the Tuatini patent and the provisional application. Since it is common practice for a later filed utility application to include more or different subject matter than its earlier provisional application, it is unclear whether the material in Tuatini relied upon by the Examiner was actually present in Tuatini's provisional application. **In fact, a review of Tuatini's provisional application reveals that it varies greatly from the Tuatini patent.** It does not appear that all of the subject matter on which the Examiner is relying on to reject Applicants' claims is also present in Tuatini's provisional application. Applicants' request the Examiner to point out the specific page and lines numbers of Tuatini's provisional application that include all of the subject matter of Tuatini's patent used by the Examiner in the rejection. Unless the Examiner can make this showing, the rejection is improper. *See, In re Wertheim*, 209 USPQ 554 (CCPA 1981).

Moreover, the Tuatini patent is not entitled to the December 30, 1999 date as a prior art date unless at least one claim of the Tuatini patent is supported (under 35 U.S.C. § 112) in the provisional application. Under 35 U.S.C. 119(e)(1), a patent is not entitled to its provisional application's filing date as a prior art date unless at least one claim of the published utility application is supported (per 35 U.S.C. § 112) in the provisional application. From a review of Tuatini's provisional application, it is not clear where complete § 112 support for a claim of Tuatini's patent is found. The rejection is improper unless the Examiner can show that Tuatini's published application has the necessary claim support in the provisional application to be entitled to the provisional application's filing date as its prior art date. *See also* M.P.E.P. § 2136.03(III, IV).

Furthermore, as shown below, Tuatini does not teach all the limitations of Applicants' claims.

Regarding claim 1, contrary to the Examiner's assertion, Tuatini fails to disclose a compilation process of a virtual machine converting a first computer programming language object into a data representation language representation of the first object; wherein the data representation language representation of the first object is configured for use in generating a copy of the first object. Tuatini teaches an application framework for developing applications including action and view handlers. Tuatini's action handlers implement business logic while view handlers control the formatting of the results returned by the business logic. Tuatini's application framework receives service requests from client computers and invokes appropriate action handlers to service the requests. The view handlers format responses and send responses to the requesting clients. (*see*, Tuatini, Abstract and paragraph 43). However, Tuatini fails to teach a compilation process of a virtual machine converting a first object into a data representation language representation of the first object, wherein the data representation language representation of the first object is configured to use in generating a copy of the first object.

The Examiner cites paragraph [0079] of Tuatini and argues that Tuatini discloses that XML is convertible to and from JAVA objects. The Examiner's interpretation of Tuatini's teachings is clearly incorrect. The cited passage actually describes generating JAVA objects to represent and manipulate the data in XML messages and regenerating the XML messages from the data in the JAVA objects. **There is no doubt whatsoever that the XML data from request messages are not *representations of JAVA objects.*** Instead, Tuatini's Java objects allow for the programmatic manipulation of the data from the XML messages. Tuatini teaches that request messages are read in and translated from a client format to an application format and that responses are translated into the client format and written out as XML messages (page 3, paragraph 0049-0050). Tuatini teaches how his deserialize method is passed an indication of the client format [of the message], the message to be translated, and an indication of the application format. Tuatini's system then "deserializes the message and returns the JAVA object *representing the message*" (emphasis added) and "performs any processing necessary to convert the message from the client format to the application format" (Tuatini, page 10, paragraph 0083-0084). Thus, any XML response message serialized from one of Tuatini's objects is not a data representation language *representation of a computer programming language object*, but instead is response message in a client format converted from a JAVA object in an application format. There is no mention in Tuatini that a XML message regenerated from an object is a representation of the object. Instead, Tuatini's system converts messages between formats to enable communication across platforms and between new and legacy applications.

Nor is Tuatini's XML response message configured for use in generating a copy of the JAVA object. Tuatini teaches that the data in the XML message is in a different format from the data in the JAVA object and thus is not configured to use is generating a copy of the object. Tuatini does not teach a data representation language representation of a first computer programming language object wherein the data representation language representation is configured for use in generating a copy of the first object. **The XML message in Tuatini is not used to generate a copy of a Java object represented by the message.**

In the Response to Arguments section of the Final Action, the Examiner states that the term "representation" is broad. However, claim 1 does not recite merely the term representation. Instead, claim 1 specifically recites a representation *of a computer programming language object* and that the data representation language *representation of the first object is configured for use in generating a copy of the computer programming language object*. Also, the term representation has a specific meaning. The XML messages in Tuatini do not *represent* a computer programming language object and are not used to generate a copy of the object.

Applicants assert that the section 102 rejection of claim 1 is not supported by the cited prior art because a rejection under section 102 requires that the **identical** invention must be shown in as complete detail as is contained in the claims and also requires the presence in a single prior art reference disclosure of each and every element of the

claimed invention, arranged as in the claim. (M.P.E.P. § 2131). See also *Lindemann Maschinenfabrik GmbH v. American Hoist & Derrick Co.*, 221 USPQ 481, 485 (Fed. Cir. 1984). Thus, for at least the reasons given above, the rejection of claim 1 is not supported by the prior art and removal thereof is respectfully requested. Remarks similar to those above regarding claim 1 also apply to claims 25, 40, 62, 71 and 84.

In further regard to claim 25, Tuatini additionally fails to disclose generating a message in the data representation language, wherein the message includes the data representation language representation of a computer programming language object; sending the message to a second process; and the second process generating a copy of the computer programming language object from the data representation language representation of the object included in the message.

The Examiner again cites paragraphs 0079 and 0082 – 0084 of Tuatini. However, as discussed above regarding the rejection of claim 1, Tuatini fails to disclose a data representation language *representations* of computer programming language objects. The Examiner's cited passage cited describes generating JAVA objects to represent and manipulate the data in XML messages and regenerating the XML messages from the data in the JAVA objects. **Nowhere does Tuatini disclose generating a copy of the computer programming language object from the data representation language representation of the object included in the message.** Instead, Tuatini converts XML messages from a client format to an application format. In other words, the XML messages in Tuatini are not and do not include representations of computer programming language objects. Generating JAVA objects to represent and manipulate data in an XML message and regenerating the XML message from the JAVA object is not the same as, nor does it anticipate, generating a copy of a computer programming language object from a data representation language representation of the object.

Furthermore, Tuatini does not disclose generating a message in the data representation language, wherein the message includes the data representation language representation of the computer programming language object and sending the message to a second process. Instead, Tuatini teaches that the server receives a request message in XML from a client; translates the XML data from a client format into an application format using a JAVA object; generates results from the request; generates a response message in XML formatted according to the client format; and sends the response message back to the client (Tuatini, page 2, paragraph 0043 and page 3, paragraphs 0049-0050). The XML based request messages sent from clients to servers in Tuatini's system are request messages that do not include data representation language representations of the computer programming language objects. Just because Tuatini teaches creating a JAVA object in response to a translated XML request message does not mean that the request message was a representation of the object. In other words, rather than being representation of programming objects, Tuatini's XML request messages are simply requests for services.

Tuatini also fails to disclose the second process generating a copy of the computer programming language object from the data representation language representation of the

object included in the message. Instead, as noted above, Tuatini only teaches generating JAVA objects that represent converted or translated versions of client request messages. Tuatini does not teach that the client send representations of computer programming language objects, but instead only teaches that client send requests for services.

For at least the reasons given above, the rejection of claim 25 is not supported by the prior art and its removal is respectfully requested. Remarks similar to those above regarding claim 25 also apply to claims 50, 62 and 84.

The Examiner also rejected claims 1-90 under the judiciary created doctrine of obviousness-type double patenting as being unpatentable over claims 1-66 of co-pending Application No. 09/663,564. The instant application and the 09/663,564 application are both pending patent application, not issued patents. If and/or when this rejection becomes non-provisional, Applicants will consider filing a terminal disclaimer or present reasons traversing the rejection.

In light of the foregoing remarks, Applicants submit the application is in condition for allowance, and notice to that effect is respectfully requested. If any extension of time (under 37 C.F.R. § 1.136) is necessary to prevent the above referenced application from becoming abandoned, Applicants hereby petition for such an extension. If any fees are due, the Commissioner is authorized to charge said fees to Meyertons, Hood, Kivlin, Kowert & Goetzel PC Deposit Account No. 501505/5181-72000/RCK.

Also enclosed herewith are the following items:

☒ Notice of Appeal

Respectfully submitted,

/Robert C. Kowert/
Robert C. Kowert, Reg. #39,255
ATTORNEY FOR APPLICANT(S)

Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8850
Date: December 18, 2006